for displaying the data to the user 36. In the preferred embodiment, a DD 20 has the form of window on the display device 16 with a hierarchy of display panes. Each pane is used to display one or more or the PRTs that comprise a MRT. The present invention also includes procedures for entering data changes using DDs. As will be described, the process for modifying data with DDs is very simple for the user. A change entered with a DD automatically updates the MRTs to update other related MRTs and updates and create XML documents.

The present invention interfaces with the underlying XML documents by copying the XML data components into normalized data objects referred to as PRTs and organizing the PRTs into recognizable business objects referred to as MRTs 24. Examples of business objects that are modeled as MRTs with the present invention include, but are not limited to: invoices, bills of material, purchase orders, price books, forecasts, and fund transactions. The present invention advantageously works in conjunction with underlying data sources 38, 40 to reconstitute data stored therein into a structure recognizable by and easily manipulated by the businessperson. The user can define functions (calculations), filters (selection criteria), sorts, and DDs (display and organization rules) over MRTs.

A MRT is a user-defined collection of PRTs organized into a hierarchy. One and only one of the PRTs uniquely define a node in the hierarchy. A single instance of a PRT results in a single instance of a MRT hierarchy node. A single instance of the PRT assigned to the top hierarchy node results in a single instance of the MRT. (An instance of an MRT is a Management Record Pointer Family (MRPF) 90. In the example of FIG. 3, the Order Header is the PRT that uniquely defines the MRT. A PRT is where the invention stores the data extracted from XML documents; PRTs are similar to relational database tables. A "hook" 44 refers to the method used for linking or showing relationships between records stored in different PRTs. Such hooks include pointers and foreign keys. A hook typically provides a one to many relationship between a parent record and a child record. As shown in FIG. 3, the hooks are represented by lines where a single occurrence of a connecting line attaches to the parent, and multiple occurrence of the connecting line attach to child. A hook not only specifies which fields of a PRT point to which fields of another PRT, but also specifies the number of hooks any particular instance of one PRT can and must have with the

10051701.072502

instances of the other PRT.

It should be understood that PRTs are unlike XML components. First an XML component instances are not shared with other XML document instances. In other words, two XML documents may have identical values in all its elements. Unlike XML components, no two PRTs have identical values in the fields that uniquely identify a PRI. Second an XML component type has at most one parent component type where as a PRT can have multiple parent PRTs. It should be understood that MRTs are different than XML documents. First, a MRT can be comprised of multiple XML documents; for example a MRT can incorporate the data in both an Order XML document, and Customer XML document, and Product XML document. Second unlike an XML document, a MRT does not have any redundant data. For example an Open Orders XML document would contain duplicate customer information for every order for the same customer and duplicate product information on every order line for the same product, A MRT does not actually include the customer and product data but has a pointer to the single instance of data for each customer and each product. MRTs require less storage space than XML documents and when users change the product information and it is immediately reflected in all MRTs that point to the changed product.

10510107 . 072560

MRTs also have significant advantages XML documents elements, and object database composite objects. A system administrator predetermines hierarchies of XML documents and object-oriented databases, typically. The users manually maintain the pointers between the individual records. Family trees (which do not show spouses) are examples of hierarchies. The relationships are rigid and only movement from parent to child and visa versa is permitted. Any new hierarchy is empty until parent and child records are explicitly linked to each other either manually or by writing a custom program to build the links. MRTs, on the other hand, automatically populate a hierarchy of PRTs using PRT hooks 81 according to definitions set by the user.

Also, unlike in an XML document component hierarchy where relationships are rigid, a parent PRT can be the child node in a MRT hierarchy. For example in FIG. 3, customer order header is typically a child component to a customer component but with the present invention, the user can define a customer order MRT where the customer order header is the parent. In this example, the customer order header is the top node for this MRT while the customer is a single occurrence child.

FIG. 3 illustrates an example of the pointer structure of the present invention that is used with the MRT 24. The pointer structure is preferably used to improve the speed for processing filters, sorts and functions. As has been noted above, one PRT 52, in this case order header, is used to uniquely identify the MRT 24. Each PRT 26 is linked either directly or indirectly by hooks 44 to the order header PRT. The present invention also produces pointer families 46-49 to divide the PRTs 26 in the MRT 24 into groups. Each of the management record pointer families (MRPF) has a lead PRT 50 that directly or indirectly through another lead PRT 50 has a many to one relationship with the unique identifying PRT 24. The PRTs that have a one to many relationship 44 with a lead PRT 50 or the one PRT 52 are included in the that PRT pointer family 46-49. The example of FIG. 3 will be used below to illustrate the operations performed by the present invention on MRTs 24.

Referring now to FIG. 4, an overview of the preferred method of the present invention is described. The preferred method begins with a series of steps in which the user inputs defining information described in FIG 4A. In step 80, the user defines or imports the XML format. In step 81 the invention creates PRTs by identifying the XML format components that have at least one child component. When the PRT already exists, the user maps the XML format components to the data elements of the already existing PRT.

In step 82, the invention creates hooks between every PRT that it creates and the XML component and the PRT it creates for its immediate parent XML component. The user can also define hook definitions between two PRTs by specifying the fields in the child PRT that corresponds to the unique identifying fields of the parent PRT. Next in step the invention creates the MRT definition and its Management Record Pointer Family (MRPF) as described in Figure 5. To manually create a MRT, the user selects a first PRT. After the user selects the PRT, the system presents the user with a list of all PRTs that have a hook with the first PRT or any PRTs in the list. The user then selects any of the listed PRT for inclusion in the MRT. After the user has completed selection of PRTs, the user specifies which one of the PRTs selected is the unique identifier for this MRT.

09767510.011501